# ホモロジーの摂動定理 (超局所解析とその周辺)

AUTHOR(S):

丸山, 文綱

CITATION:

丸山, 文綱. ホモロジーの摂動定理 (超局所解析とその周辺). 数理解析研究所講究録 2000, 1158: 17-25

ISSUE DATE:

2000-06

URL:

http://hdl.handle.net/2433/64199

数理解析研究所講究録
1158巻 2000年 17-25



# ホモロジーの摂動定理

丸山文綱(MARUYAMA Fumitsuna)

## 0. 導入

奴隷にならないための数学が重要であることはいまさら私が繰り返して言うことでもないが、奴隷にさせるための数学も現状からみてそろそろ本腰を入れて考えられるべきであろう。後者の奴隷とは偉大なる石頭、計算機のことである。計算機は人間の命令をそのまま高速に実行してくれるというありがたい特性があるが、逆に人間がちゃんと指示してやらないと出てくる結果は意味のないものとなる。すでに数学をするためのソフトウェアは数多くあるが、ここではフランスの数学者 Sergeraert によるソフトウェアについて簡単に述べることで、計算機による数学の問題点などを見ることにする。

出てくる話には特別新しいものがない。しかし、Sergeraert が日本に十年程前に来ていくつかの大学で講演しているにもかかわらず、代数的位相幾何の人の興味しか引かなかったようで、意外に知られていないようなので紹介することは無駄ではない

と思われる。彼の論文、ソフトウェアはFourier研究所のWEBページ

http://www-fourier.ujf-grenoble.fr/~sergerar

から入手可能である。

## 1. 数学から計算機科学へ

二十世紀の数学にとって大きな意味を持つ結果としてGödelの不完全性定理がある。基礎論はこの時点で大きな二つの方向に分かれ、計算機科学につながる。Turingらの機械による論理と、Churchらの帰納的関数（＝アルゴリズム）の定義である。前者の流れは、いわゆるプログラム内蔵型(von Neumann 型)計算機の誕生、それに続くFORTRAN(FORmulae TRANslator)、Cなどの手続き型(Procedure)言語の現在まで続く隆盛を招くこととなった。後者の流れは関数型(Functional)言語 LISP (LISt Processor) を生む。

LISPは最も古い計算機言語のひとつでありながら、計算機言語を分類する際に、「LISPとそれ以外」といわれるほど独特な言語である。もちろん方言も色々あるが、共通して言えることはリストと呼ばれる再帰性を表現するのに最適な構造を（ある意味でそれだけを）扱うということである。リストでは自然に集合を表現することもでき、プログラムの入力、出力の数に関

して注意を払わなくてもよいようにできる。関数型とはプログラムが関数のように働くことを意味する。(ちなみに、このリスト構造をもとにしたLOGOという言語が作られ、幼児教育等に使われたことがある。この発案者 Papert が提唱したのがMINDSTORMS という概念で、これを受け継いで例の大人のおもちゃができたわけである。) LISPの特徴としてデータとプログラムの区別がないことがあげられる。また記号を扱うことが楽なため、人工知能などに使われ、数学に関係したところではREDUCEという数式処理プログラムが開発された。

## 2. 数式処理

数式処理とは、たとえば不定元を不定元のまま計算(表象計算)することである。計算機というと数値計算ばかりが思い出されるようであるが、こういうことができないと一般的には数学に使えない。取り扱う対象が多項式の計算だけならプログラムを組むことは実は意外に簡単であるが、特殊関数などが入ってくるとその時点でその関数に関する「知識」を覚えさせることが必要になる。そしてそういう知識、つまり定義式、恒等式や、特殊な値を組み合わせて使い、式を簡約化することができなければならない。(もちろんプログラムに何を求めるかにもよるが。)たとえば簡単な二次方程式の根でさえ、根号がきちんと

扱えなければ数学的に正しく表示できない。

ここで数学と計算機の大きな差について述べておかなければならない。それは無限についてである。数学には無限の概念が自然に出てくるが、計算機はそのままの形では無限を扱うことができない。記憶容量の制限の問題もあるし、時間が無限に使えるわけでもないからである。ただし、似たような状況は人間にもあるわけであるから、人間がどうやっているかを参考にすることにより、扱うようにできないわけではない。つまり数学記号Rが実数の集合を表すように、記号化すればよいのである。ただし、この記号化によってやはりその実体に関する知識を導入してやらねばならない。上に例を出した特殊関数や根号も、無限級数の性質を抽出したり、無理数の持つ無限桁を有限に表示したりする手段であるとも言える。

さらに一般には有限でさえ限られた形でしか扱えない。たとえば準備がないと 100!のような数を具体的に算出することはできないのである。このような多倍長精度演算ができることが数学用ソフトウェアとしては必要になってくるのはあきらかである。実は多項式の計算でも、たとえばBuchbergerアルゴリズムによるGröbner基底の計算途中に係数が爆発的に大きくなる例が知られており、決して大きな数を取り扱うことが目的でなくても、自然に大きな数は出てくることがあるからである。

以上のようなことを踏まえて考えると、一般的に記号処理が数学ソフトウェアにとっていかに重要であるかがわかる。しかしだからといって必ずしもLISP言語以外はよくないというわけではない。実際LISPには利点の裏返しの欠点もあり、その上現状では計算機のほとんどがプログラム内蔵型である（つまりLISPそのものがその上で動いている）ので、C言語によって作るメリットは十分にある。

## 3. Sergeraertのソフトウェア

実はSergeraertのソフトウェアは前節で述べたような一般的な計算をするソフトウェアではない。代数的位相幾何学（Algebraic Topology）の限られた問題、ループ空間のZ係数ホモロジーを計算するためのものである。彼はこの手法を実効代数的位相幾何学（Effective Algebraic Topology）あるいは構成代数的位相幾何学（Constructive Algebraic Topology）と呼んでいる。実際彼のプログラムの初期バージョンの名前はEATであり、昨年発表された新しいバージョンは彼の猫（つまり英語でCAT）の名前からKENZOと名づけられている。

ではホモロジーの計算での問題点を挙げてみよう。まず、群や環など数学的構造の計算機上での実現であり、その計算である。また無限の取り扱いも問題になる。これらの実現のために

彼はLISP上にシステムを構築した。数学的構造をLISPのプログラムかつデータとして扱うことで、数学的構造のもつ具体的内容と対象としての記号化を同時に実現している。彼はこれをFunctional Algorithmicと呼んでいる。圏の理論をそのまま計算機上に持ち込んだようなものと考えればよい。これによってスペクトル系列の計算が（実効的に）可能になった。

また、(コ)ホモロジーに付き物の輪状部分に対しては、摂動定理（Perturbation Lemma）によって簡約化している。この定理を使う数学的背景としてRubioによるAdamの問題の解決があるが、ここでは説明しない。この定理と上に述べた記号化をあわせることによって非自明な対象のホモロジーが計算できるようになったわけである。

## 4. ホモロジーの摂動

この節ではこのホモロジーの摂動というちょっと見には驚かされる内容の定理の内容を紹介する。まず、ホモトピーによる同値関係を構成するために、次の定義をする。

定義

次の組 $(\tilde{C},C,f,g,h)$ を REDUCTION と呼ぶ。

$$\tilde{d}\circlearrowright\tilde{C}\circlearrowright h$$
$$f\downarrow\uparrow g$$
$$d\circlearrowright C$$

ただし d（resp.$\tilde{d}$）は鎖複体 C（resp. $\tilde{C}$）の微分写像で、f,g は複体間の写像、h はホモトピー写像（つまり微分 $\tilde{d}$ とは逆方向の $\tilde{C}$ 上の写像）で、次の(1)～(3)を満たすものとする。

(1) $f\tilde{d} = df$ $\qquad$ $\tilde{d}g = gd$

(2) $fg = id_C$ $\qquad$ $gf + \tilde{d}h + h\tilde{d} = id_{\tilde{C}}$

(3) $fh = 0$ $\qquad$ $hg = 0$ $\qquad$ $hh = 0$

ここで $id$ は恒等写像である。

このとき、特に Z 自由加群の場合 $\tilde{C} = Ker\, f \oplus \mathrm{Im}\, g$ であり、後者は $C$ そのものに同型である。

<u>定義</u>（ホモトピー同値）

ふたつの REDUCTION の組 $(\tilde{C}, C_1, f_1, g_1, h_1)$ $(\tilde{C}, C_2, f_2, g_2, h_2)$ があるとき、$C_1$ と $C_2$ をホモトピー同値と呼ぶ。

$C_1$ と $C_2$ のホモロジー群が等しいことは明らかである。

<u>定理</u>（**Perturbation Lemma ; Shih, R. Brown, etc.**）

REDUCTION（$\tilde{C}$, C, f, g, h）ただし d（resp.$\tilde{d}$）は鎖複体 C（resp. $\tilde{C}$）の微分写像とする）に対して、$\tilde{C}$ の写像 $\tilde{\delta}$ で次を満たすものがあるとする。

(1) $(\tilde{d} + \tilde{\delta})^2 = 0$

(2) $\tilde{C}$ のすべての元 $x$ に対して

$$\lim_{n\to\infty} (h\tilde{\delta})^n x = 0$$

$$\Leftrightarrow \quad \lim_{n\to\infty} (\tilde{\delta}h)^n x = 0$$

（つまり $\tilde{d}+\tilde{\delta}$ が微分で $h\tilde{\delta}$ および $\tilde{\delta}h$ は冪零。）

このとき C の写像 $\delta$ と REDUCTION $((\tilde{C},\tilde{d}+\tilde{\delta}),(C,d+\delta),f',g',h')$ が存在する。

証明

$\delta,f',g',h'$ を具体的に構成すればよい。

$$\Phi=\sum_{j\geq 0}(-1)^j(h\tilde{\delta})^j$$

$$\Psi=\sum_{j\geq 0}(-1)^j(\tilde{\delta}h)^j$$

とおく。すると

$$\Phi=id-h\Psi\tilde{\delta} \qquad \tilde{\delta}\Phi=\Psi\tilde{\delta}$$

$$\Psi=id-\tilde{\delta}\Phi h \qquad \Phi h=h\Psi$$

である。これらを使い、

$$f'=f\Phi \qquad g'=\Psi g$$

$$h'=\Phi h=h\Psi \qquad \delta=f\tilde{\delta}\Phi g=f\Psi\tilde{\delta}g$$

とすればよい。REDUCTION の条件はすべて満たされていることがわかる。

注：証明をよく見ると（Neumann 級数が見えていることからも）わかるとおり、この定理は陰関数定理の応用に過ぎない。しかし、一般的な（コ）ホモロジーを計算するのにこの摂動定理が使えるかというと、そう簡単ではない。ホモトピー写像および

摂動写像がうまく見つかるかどうかが鍵になるわけで、その存在を数学的に裏付けておかないといけないわけである。

## 5. 最後に

特に新しいことはないと思うので、参考文献は紹介しない。はじめに紹介した WEB ページなどから必要な情報は得られるはずである。

数学のソフトウェアについては、ハードウェアについての制約が少なくなってきた今こそ、より多くの数学者を巻き込んで議論されるべきときであると思う。すでに Gröbner 基底については、この考究録などを見てもわかるとおり、かなり議論されている。何がしたいのか、どうすれば計算できるのかなどの情報を集積することでプログラムを作ることができる。なんだ、そんなこともできないのか、という批判がプログラムを進化させる。結果として、数学的実験と呼んでよい面倒な計算が手軽にできることが、数学の今後の発展につながると信じている。